[SVN trunk(主线) branch(分支) tag(标记) 用法详解和详细操作步骤](http://blog.csdn.net/vbirdbest/article/details/51122637)

标签： [SVN trunk主线 branch分](http://www.csdn.net/tag/SVN%20trunk%e4%b8%bb%e7%ba%bf%20%20branch%e5%88%86)

2016-04-11 17:02 10892人阅读 [评论](http://blog.csdn.net/vbirdbest/article/details/51122637#comments)(1) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/vbirdbest/article/details/51122637#report)

 分类：



工具（1）



版权声明：本文为博主原创文章，未经博主允许不得转载。

目录[(?)[+]](http://blog.csdn.net/vbirdbest/article/details/51122637)

**使用场景：**

假如你的项目（这里指的是手机客户端项目）的某个版本（例如1.0版本）已经完成开发、测试并已经上线了，接下来接到新的需求，新需求的开发需要修改多个文件中的代码，当需求已经开始开发一段时间的时候，突然接到用户或测试人员的反馈，项目中有个重大bug需要紧急修复，并且要求bug修复后要立即上线；此时应该怎么修复bug呢？是在当前已经开发新需求的基础上进行修复吗？答案是否定的，原因是：如果是在已经开发新需求的基础上进行修复bug，那么新需求还没开发好，更没有测试，怎么立刻（或最可能快的）上线？！再次如果新功能的开发和bug修复的代码都涉及到同一段代码冲突了怎么办 。很显然不能在当前开发的代码基础上进行bug修复工作完美的解决方案是：在当时完成的那个版本中进行bug fix，这样带来的好处是：  
1：bug修复好之后可立即上线，不会因为新需求还没有完成或测试而延迟上线时间

2: bug修复是在原来上线的那个版本进行修复的，引起新bug的风险小，如果是在新需求的基础上修复bug, 那么新功能可能会带来新的bug

**SVN仓库目录结构Repository：**

trunk  
tags  
branches

**trunk(主干|主线) branchs(分支) tags(标记)**

truck(主干|主线|主分支)：是用来做主方向开发的，新功能的开发应放在主线中，当模块开发完成后，需要修改，就用branch。  
branch(分支)：分支开发和主线开发是可以同时进行的，也就是并行开发，分支通常用于修复bug时使用

tag(标记)：用于标记某个可用的版本，可以标记已经上线发布的版本，也可以标记正在[**测试**](http://lib.csdn.net/base/softwaretest)的版本，通常是只读的

**SVN具体操作步骤：(TortoiseSVN版本: 1.8.8)**

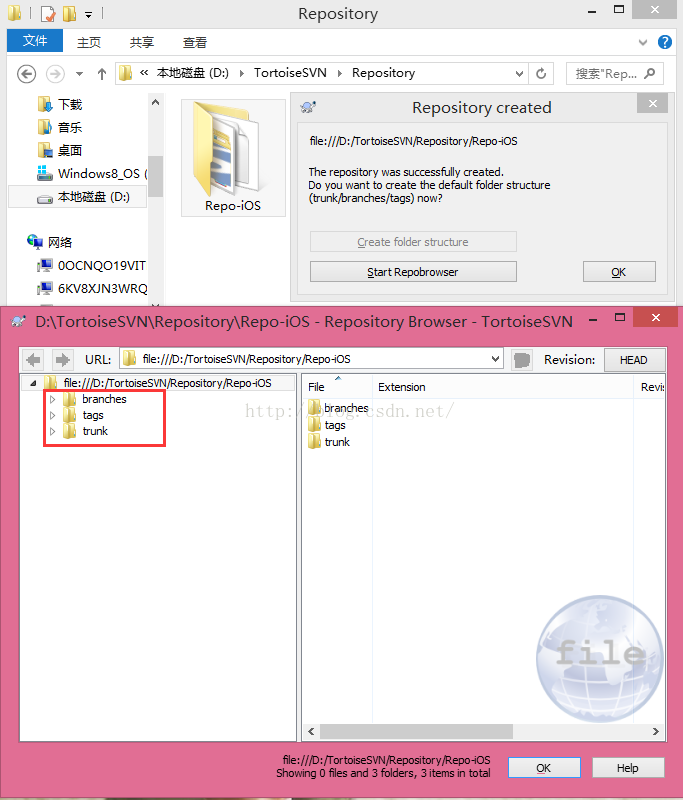
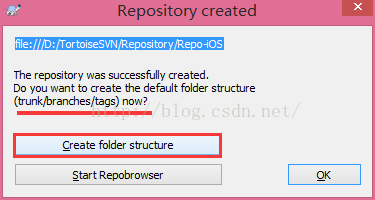
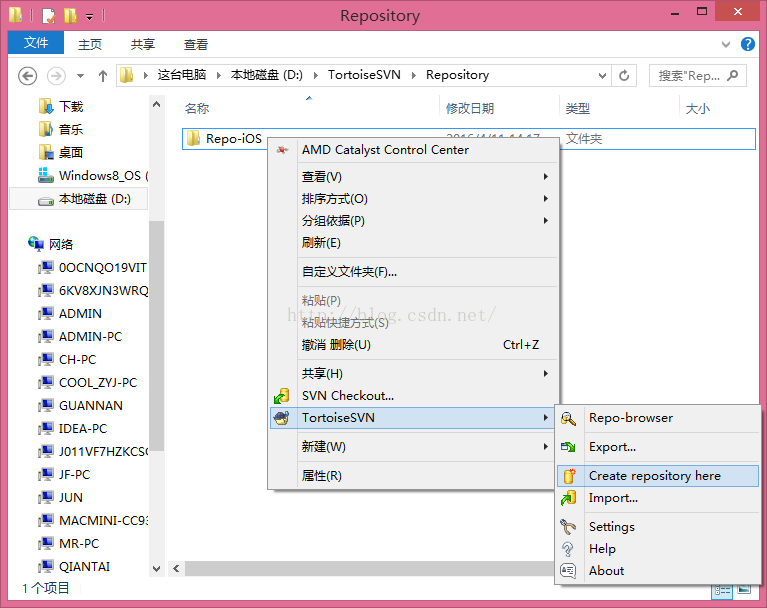
**一：创建仓库**

1. 创建目录结构D:\TortoiseSVN\Repository\Repo-iOS

2. 在该目录结构上右键

---> TortoiseSVN

---> Create repository here(创建仓库这里)   
---> Create folder structure(创建文件结构) ---> Start Repobrowser(开始仓库浏览) ---> Ok



**二：将项目上传到SVN上**

       桌面右键

       ---> TortoiseSVN

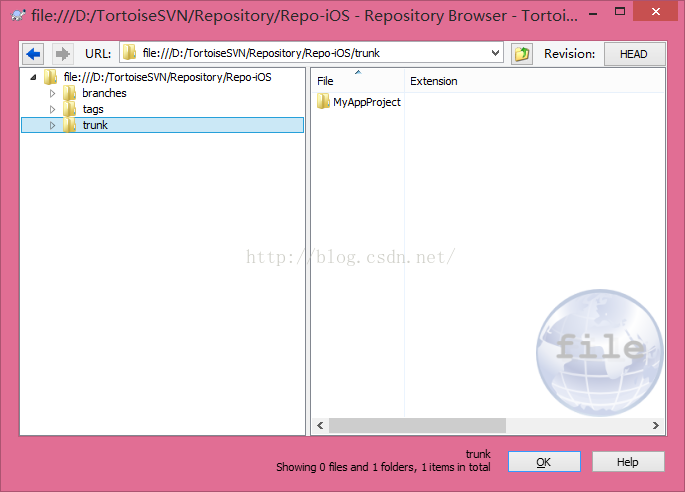
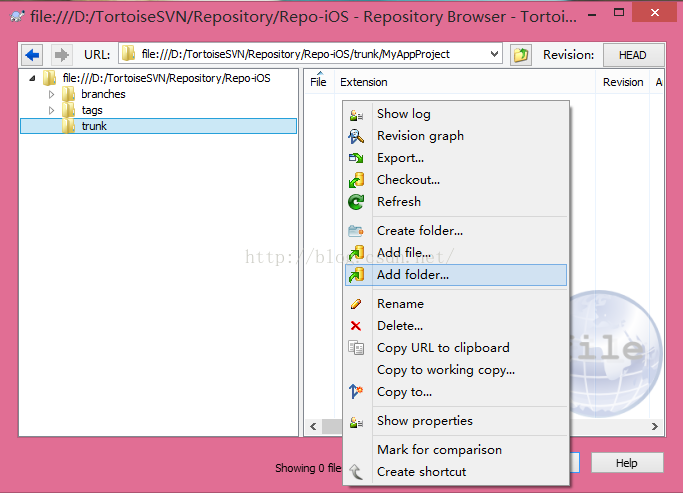
       --->repo-browser--> URL:  file:///D:/TortoiseSVN/Repository/Repo-[**iOS**](http://lib.csdn.net/base/ios) ---> Ok

       --> 选中trunk文件夹右键

       ---> Add folder...

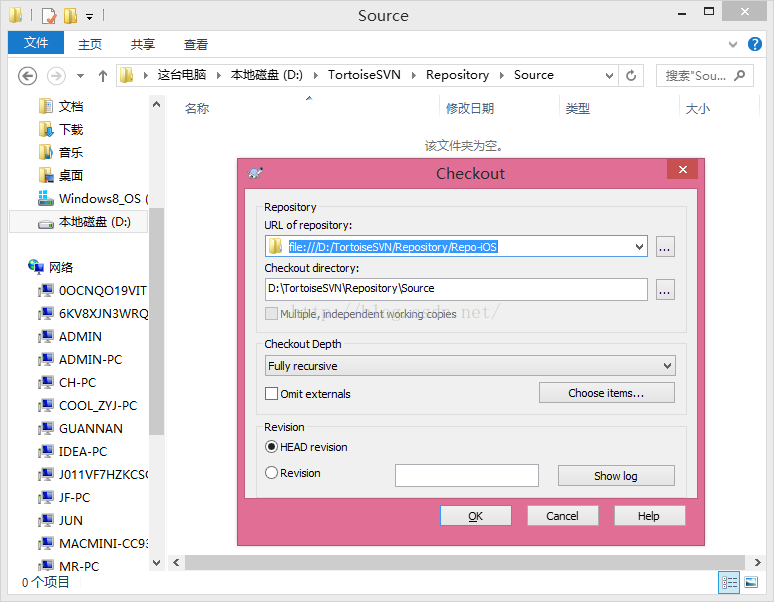
       ---> 选中要上传到SVN的项目的最外层目录，输入日志

       ---> Ok



**三：Check Out**

1. 在电脑任意位置创建一个存放项目代码的目录，例如：D:\TortoiseSVN\Repository\Source  
2. 将代码检出到该位置



**四：开发周期**

**1. 在目录D:\TortoiseSVN\Repository\Source\trunk\MyAppProject上进行开发，注意是在trunk主线上**

因为项目刚建立，这是在开发新功能，所以要在主线上开发

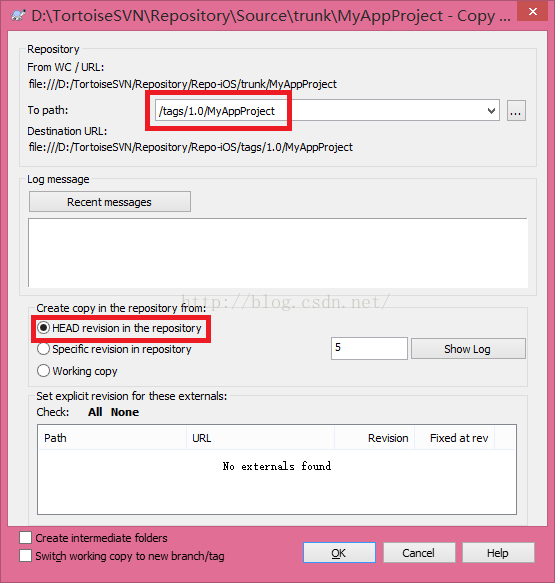
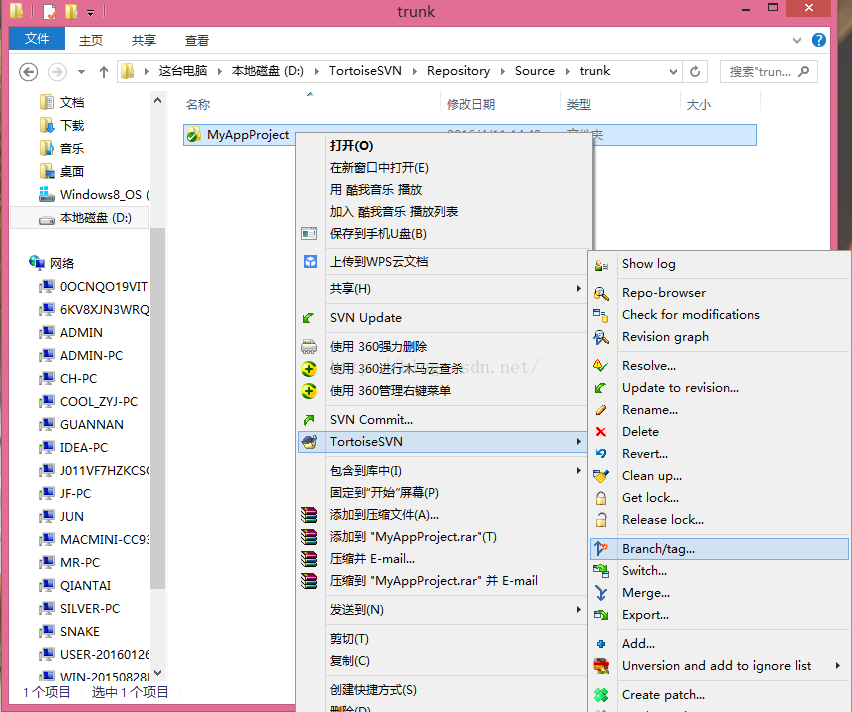
       **2. 开发一段时间后，经测试，上线到App Store，**[**Android**](http://lib.csdn.net/base/android)**上传到其它应用商城，摘取上线时的HomeViewController文件中一段代码如下**

**[objc]** [view plain](http://blog.csdn.net/vbirdbest/article/details/51122637) [copy](http://blog.csdn.net/vbirdbest/article/details/51122637)



1. - (**void**) viewDidLoad {
2. [**super** viewDidLoad];
3. // -----------------------------------------
4. **int** y = 0;
5. **int** result = 10 / y;
6. NSLog(@"iOS APP 第一阶段开发完成了！ 结果是：%ld", result);
7. // End
8. }

**3. 在D:\TortoiseSVN\Repository\Source\tags 目录下新建一个目录：1.0,并将该目录提交到SVN上，然后右键         D:\TortoiseSVN\Repository\Source\trunk\MyAppProject该目录---> TortoiseSVN---->Branch/tag... -----> To Path :/tags/1.0/MyAppProject 并选中 Head revision in repository ---> Ok此时Source/tags/1.0 目录中没有任何内容，需要更新一下该目录做update操作。更新之后看到一个完整的项目源码保存到该目录中（该目录下的源码可看做是trunk目录下版本为1.0的一个副本），查看一下/tags/1.0/HomeViewController中的viewDidLoad和trunk/MyAppProject/HomeViewController中的viewDidLoad代码完全一样。**



**4. 开发下一阶段的新需求，开发中ing**

**5. 用户或测试人员反馈应用有重大bug，需要立即修复该bug并尽快上线， 此时程序员需要为tags/1.0 下的MyAppProject 打一个分支branch，  
          操作过程如下：选中Source/tags/1.0/MyAppProject 右键 TortoiseSVN---->Branch/tag... -----> To Path ：/branches/MyAppProject  ---> Ok  
 此时看D:\TortoiseSVN\Repository\Source\branches目录下仍然没有任何内容，也需要update一下，更新之后发现该目录下  
 也出现一个完整的项目代码（该代码可看做是tags/1.0/MyAppProject的一个副本），注意打分支和打标记都是使用Branch/tag...菜单，不同的是To Path 的目录不一样，图解看打分支的图，只是to path 值不一样，此时branches/MyAppProject/HomeViewController中的viewDidLoad和tags/1.0/MyAppProject/HomeViewController中的viewDidLoad代码完全一致。**

       **6. 切换工作空间，使用Xcode|Eclipse集成工具打开/branches/MyAppProject下的项目，然后在此基础上调试并修复bug，注意打开的必须是分支中的项目**

       **7. bug 修复好后，先提交修改的文件，并进行客户端App上线，上线完成后再将branches/MyAppProject/打个tag到1.0.1目录下（tags/1.0.1）（操作步骤同步骤3），tag操作完成后，可以看到tags/1.0.1/HomeViewController.viewDidLoad 和  branches/MyAppProject/HomeViewController.viewDidLoad是完全一致的，将branches/MyAppProject打一个新的tag是以便于下次在此基础上再次修复bug，至此bug修复已经完成；修复bug后的代码如下：**

**[objc]** [view plain](http://blog.csdn.net/vbirdbest/article/details/51122637) [copy](http://blog.csdn.net/vbirdbest/article/details/51122637)



1. - (**void**) viewDidLoad {
2. [**super** viewDidLoad];
3. // -----------------------------------------
4. **int** y = 10;
5. **int** result = 10 / y;
6. NSLog(@"iOS APP 第一阶段开发完成了！ 结果是：%ld", result);
7. // End
9. NSLog(@"1.0 版本闪退bug 已修复, 程序出现除0异常");
10. }

**8. 接下来将branch和trunk进行合并，操作步骤如下：**

**右键 branches/MyAppProject ------>TortoiseSVN**

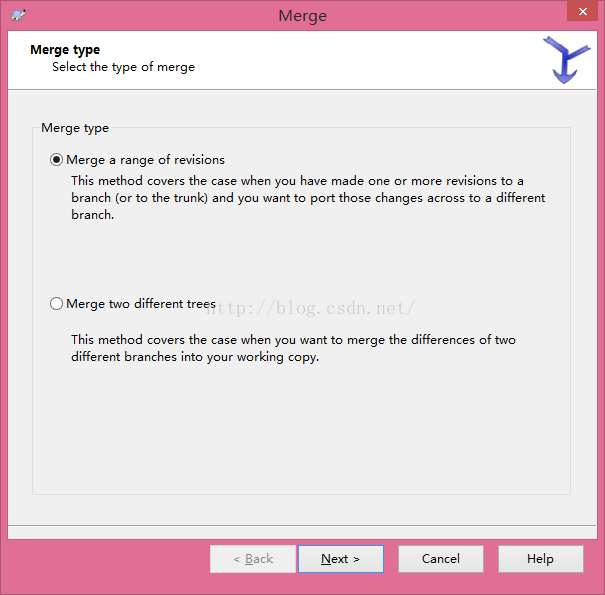
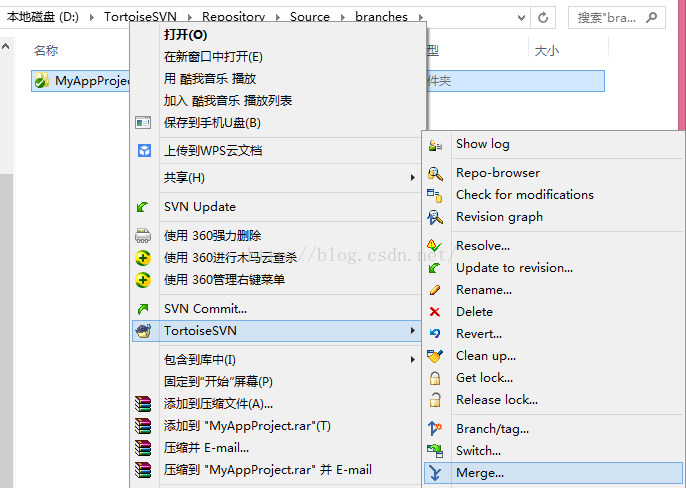
**----> Merge...**

**---> Merge a range of revisions**

**----> Next --->URL to merge from : file:///D:/TortoiseSVN/Repository/Repo-iOS/trunk/MyAppProject**

**----> Next**

**----> Merge**



**9. 将trunk和branches进行合并 步骤如下，**

**右键/turnk/MyAppProject ----> Merge... -----> Merge a range of revisions   -----> Next**

**----->  URL to merge from : file:///D:/TortoiseSVN/Repository/Repo-iOS/branches/MyAppProject**

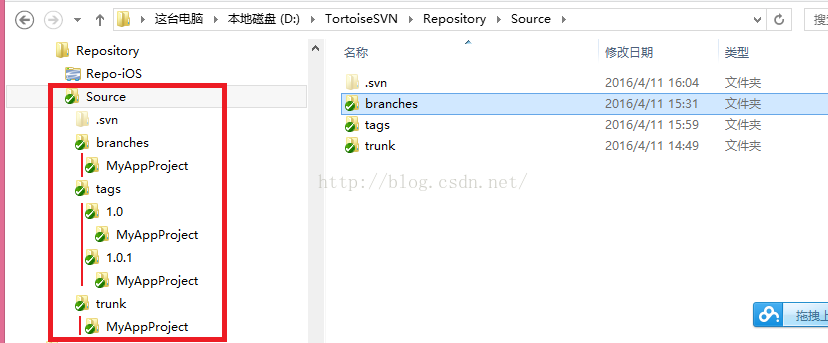
**步骤同上，只是  URL to merge from   的路径不一样。合并完成后，查看一下/trunk/MyAppProject/HomeViewController/viewDidLoad方法如下：**

**[objc]** [view plain](http://blog.csdn.net/vbirdbest/article/details/51122637) [copy](http://blog.csdn.net/vbirdbest/article/details/51122637)



1. - (**void**) viewDidLoad {
2. [**super** viewDidLoad];
3. // -----------------------------------------
4. **int** y = <**strong**>10</**strong**>;            // <strong>可以看到branches分支中的代码已经合并到主线上了</strong>
5. **int** result = 10 / y;
6. NSLog(@"iOS APP 第一阶段开发完成了！ 结果是：%ld", result);
7. // End
9. NSLog(@"<strong>1.0 版本闪退bug 已修复, 程序出现除0异常</strong>");
11. NSLog(@"其他同事在主线trunk中进行新需求开发...");
12. NSLog(@"其他同事在主线trunk中进行新需求开发...");
13. }

**10. 此时合并彻底结束，branches目录下的源码如果不想要也可以删掉，接着修改bug的这位程序员需要切换工作空间到主线上来，使用Xcode重新打开trunk/MyAppProject项目，接着开发尚未完成的新功能。SVN目录如下：**



**说明：**

**1. 分支开发和主干开发是两个完全独立的过程，两者可以同时进行开发**  
**2. 因分支和主干开发是并行的，所以两者可以任意的多次提交当前工程所修改的文件**

本人也是刚接触trunk，上面描述如有问题，欢迎大家评论指正